Common LabVIEW Conventions

The data-taking and some of the data-analysis programs are written in LabVIEW, a language which provides a lot of useful user-interface elements without much work on the part of the programmer. Hence, programs written in this language tend to have a common look and feel somewhat different from the usual Windows or Mac programs you might be familiar with. If you're not familiar with LabVIEW, then you will find it helpful to read this description of how to use the common GUI elements of the language.

LabVIEW tries, with varying amounts of success, to make programs shown on a screen look and feel like physical instruments covered with knobs, switches, dials, etc. Since there are no physical objects present, you have to use the mouse to indicate what you want to do. Thus, to turn a knob, you click and drag on a point on its perimeter. The analogy is used to such an extent that all LabVIEW documentation refers to the user interface of a program as its 'front panel'. Programs also have 'block diagrams' which constitute their source code.

One of the fundamental concepts is that of controls and indicators. A control is a way you tell the program what you want it to do or the value of a quantity. An indicator is a way the program tells you something. Thus, a knob is usually a control. However, all controls can be loaded with values and used as indicators, and all indicators can be set up to be used as controls. Confusing? Yes, but it usually works out in an intuitive way. For instance, graphs are indicators, but they can have cursors on them, which can be used as controls. Another example is LEDs. These are used for Boolean (on/off) indicators, showing bright when TRUE/ON and dark when FALSE/OFF. However, they're often used as pushbuttons for selecting things. For instance, the Plot Specification controls include arrays of LEDs. These LEDs are initially set to register as specified in a file, but the user gets to manipulate these and change their states. Similarly, other sorts of controls are often pre-loaded with default information so you don't always have to set them.

While on the subject of Booleans, I'll mention a convention with regard to switches. These are controls or indicators which look like toggle or slide switches or fancy 3D switches with lights. You work these by clicking on them; you don't click and drag as you do with knobs. The convention is that ON is represented by a switch that is up or to the right, just as on physical devices.   Sometimes these switches will have labels ending in a ?, like "Delete all data at end of run?". In these cases, up or right means yes.

It can be hard to tell if a button is 'pushed', i.e. on. The convention is that the 'light' comes from the upper left, so a button which is not pushed, hence 'raised', will have a bright arc from about 7:30 to 1:30 along its circumference and a dark arc the rest of the way around, while these colors will be reversed if the button has been 'pressed'. Many buttons 'pop out' as soon as they have been pressed and that action sensed by the program. If such a button remains 'pushed', then there may be something wrong.

Some controls have a little box off to the side with an integer showing, usually 0. These controls are array controls which specify or indicate lists of quantities. The LEDs in the Plot Specification controls are array controls. There may be more LEDs showing than there are elements defined in that array. In that case, the excess lights are 'grayed out' (see below). The index display has a little spinner by which you can scroll up or down the list, and you can also type in a number. Thus, if there are 15 elements in an array indicator and you want to see the 25'th, you can type 15 into the index display and then the desired element will show up as #10.

That brings to mind a convention which will be natural and familiar to computer types, especially C programmers, and confusing to others. That is, all lists start at the 0th element. Thus, the first scaler is scaler 0. The first element to appear in any array display is likewise #0. Some of the beamline hardware works the same way. Likewise, the first cursor on a graph is called Cursor 0 by default.

One control which doesn't have a physical analog is the File Path control. This looks like a small rectangular typing area, often with a folder icon next to it. That folder icon is a Browse button which lets you search for files of a given sort. The dialog box it's used with follows the same conventions as any standard Windows file box. You can also type directly into the white (usually) typing area.

Another non-physical control is the String control. This is a place where you can type something. For instance, in the EXAFS data-taker, energies may be specified not only as numbers like 8978 but by alphanumeric combinations such as 'cuk+50', which means 50eV above the Cu-K edge. String indicators are often used to report back to the user such information as what the current file is and any error conditions encountered.

All buttons, switches and knobs in LabVIEW can be made to do something physical knobs can't do: they can go gray and become locked out. This means exactly what 'grayed out' means in Windows. For instance, if you're in one of the editing routines and what you've specified isn't possible, the Accept button which would otherwise solidify your choice becomes grayed out. This is a clue that something's wrong and needs to be fixed before proceeding. Similarly, in the EXAFS data-taking program, you can’t move the monochromator while scanning, so the Move Mono buttons are grayed out. This mechanism is a reasonably elegant and gentle way of prohibiting attempted actions which don’t make sense or would be harmful. It beats being yelled at or having dialog boxes pop up and have to be clicked on. If the gray-out is due to an error condition, a message will appear in a box on screen to tell you what’s wrong.

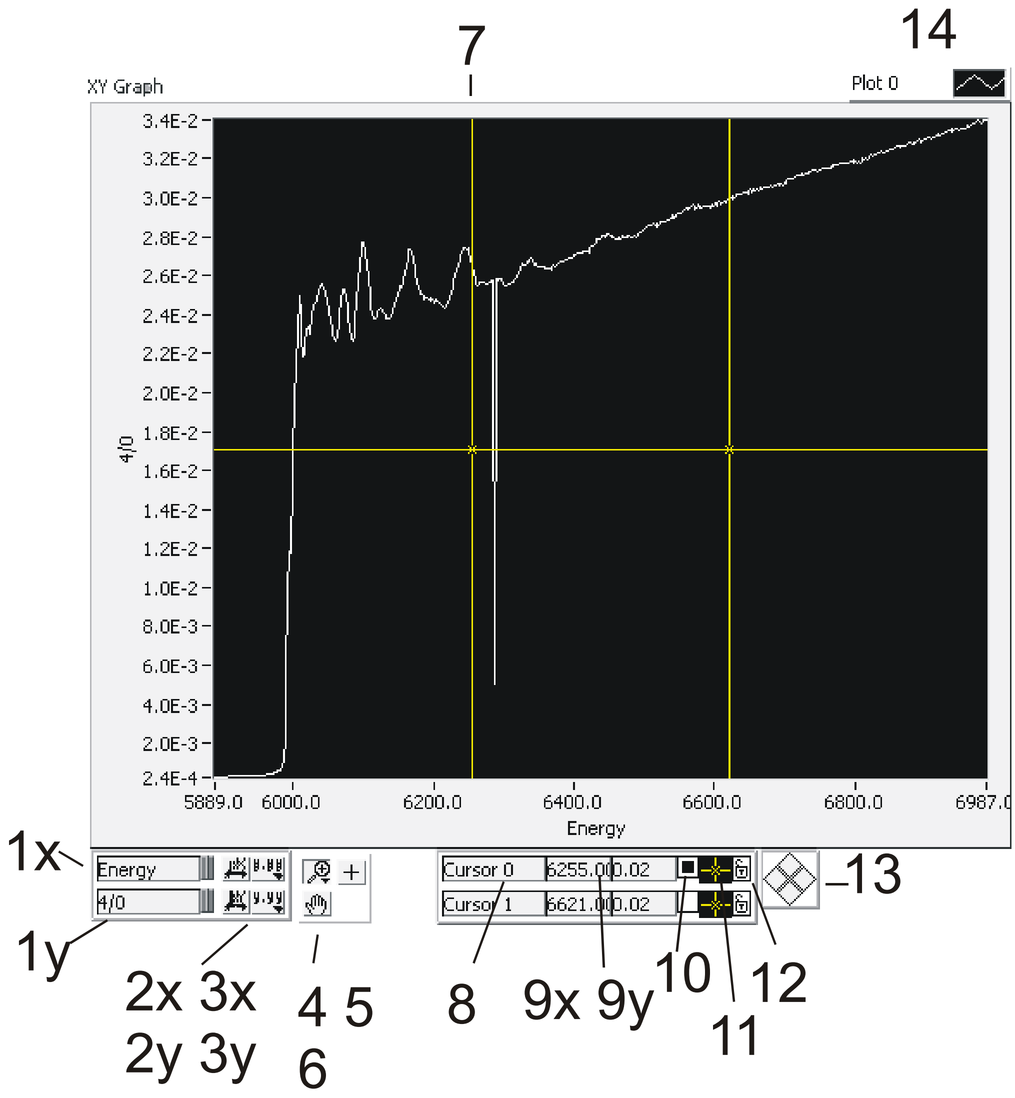
All LabVIEW programs have an arrow on a pushbutton near the upper left, just under the title bar. This is how you tell it to run. If it’s a white arrow, it means that LabVIEW is waiting to run the program. At this stage, you can do things to controls but nothing will happen. Browse boxes work, so you can preview where files are before starting out. At this stage, though, the program is vulnerable to tinkering unless it’s been protected by a password or its file has been write protected. It’s possible to change any color, size, shape or format of any display item. When you click on the white arrow, it turns black. That’s the sign that the program is running. It’s possible to stop a program using the stop-sign icon to the right of the arrow, but there’s usually a more graceful way of stopping in the form of a Stop or Exit button in the program. The stop sign is like ^C or control-Break in DOS; it’s a hard stop with no provision for cleanup. Use only if hung.

Some buttons cause a new front panel to pop up. When this happens, you are in a sub-program with its own controls and copy of the data. There will be one or more buttons marked ‘Accept’, ‘Cancel’ or ‘Return’ which let you back into the main program. This type of interface is used when the front panel of the sub-program is too complex or large to fit easily within the main screen. Think of such sub-programs as being like

Windows Wizards or the sort of menu item whose name ends in an ellipsis, such as "Options...".

Tab controls constitute a new feature in LabVIEW 6, which has been used extensively in our code. A tab control looks like a set of tabbed folders, or like the tabs in Excel or certain Windows controls (for example, Tools->Options in Word and others). Each page of a tab control can have other controls or displays on it. Thus, functions can be segregated, resulting in a cleaner front panel.

One of the most complex types of indicators in LabVIEW is the graph. There are several types of graph, but they have common features. The next page shows a typical graph and callouts explaining the features. Since tools for scaling, cursors, zooming, axis labeling and formatting, etc. come with LabVIEW graphs, we have come to rely on them instead of coding such tools into the programs. Thus, if you want to zoom on a specific area of a graph, you can do it, but you have to do it using LabVIEW's tools. Similarly, if part of a curve falls off a graph and you want to retrieve it, you need to know which button to push to do it. Thus, the next page shows a typical graph and its various parts. Not all graphs will have all these parts, and some will have more than one curve on them. Also, the background on the controls of this graph is lighter than usual in order to make the controls more obvious. This alteration is an example of what can be done to the front panel when the program is loaded but not running.



1x,y: Names of X,Y axes

2x,y: Scale-to-fit buttons for X,Y axes. Push this to make the whole curve fit in the graph.

3x,y: Scale format. Use these to adjust how the scale numbers are displayed. These pop up sub-menus which let you specify how many decimals to display and whether to use scientific notation.

4,5,6: Zoom, cursor and pan tools. When the zoom button is in, as in this example, the mouse lets you zoom in the image. The type of zoom (window, zoom in/out, zoom X only, zoom Y only, undo last zoom) may be changed by clicking on the button. Clicking on the pan (hand) tool changes the mouse pointer to a hand, which lets you move the graph around in its frame. The + is the cursor button, which lets you grab the cursor.

7. One of two cursors on this graph. These are used to select regions of interest. When a cursor is showing, you can grab it with the mouse pointer provided the + button (6) is 'pushed'. It doesn't matter which cursor you use; the program will sort out which is left and which is right.

8. Cursor name. Since it doesn't matter which is which, Labview's default names are retained.

9x,y: Coordinates of the cursor.

10: The black square shows that the top cursor is ready to be manipulated. You can switch to the other one by clicking on it or its square.

11. Cursor appearance control. Clicking on this brings menus allowing you to change the color and style of the cursor.

12. Cursor lock indicator. If this shows 'locked', the cursor is constrained to follow the graph on the screen. This feature is automatically engaged in one of the background-subtraction pop-ups in the data editor.

13. Cursor movement buttons. These arrows let you move the cursor by 1-pixel increments.

14. Plot legend. When more than one curve is plotted, shows which graph is which. For this particular graph, there is only one curve, so no effort was made to change the name from the default. Right-clicking on this brings up a menu which allows you to change the style and color of the plot, e.g. making it plot as green dots instead of a white line.